;login:
THE UNIX NEWSLETTER

Volume 6 Number 1 January 1981

CONTENTS

Notice

* UNIX is a trademark of Bell Laboratories

Guidelines for Submission of Newsletter Material

I would like to use the modern text preparation and communications facilities of UNIX to as great an extent as feasible in the preparation of the Newsletter. I have established an account on our PWB/UNIX system so that those who can provide us with machine manageable material can do so. The telephone number is (512) 474-5511. The login name is login and the password is usenix. (The system is also host utexas on the ARPANET.)

For those submitting paper copy of material, please produce your copy on a daisy wheel printer or similar high quality printing device. Line printer produced copy is typically not adequate for reproduction. Copy should be on 8 1/2" by 11" paper with a 1" margin on left, right, and bottom and 1 1/2" margin on top.

U. S. Mail submissions should be addressed to:

Login Newsletter
Computation Center
The University of Texas
Austin, Tx 78712

Attn: Wally Wedel

Letters

To: Unix People

Subject: Need for new tape archive format.

From: Dave Yost, Rand Corp.

Date: January 20, 1981

I believe that tar has enough limitations that a new and better tape archive format should come into being. What? ANOTHER tape format, I hear you ask? Why not extend tar? My answer is that tar is not extendable, and that that is only one of its problems.

My gripes with tar fall into two classes: those which can be fixed by modifying the tar program while still using the same tape format, and those which will require a different tape format. For now, I will limit the discussion to the latter.

Gripes:

* User and group id's are stored as numbers. Extracting the same tape on different systems would be easier in many situations if they were stored as strings.

* File names are fixed-length.

* Only the modtime of each file is stored.

* Multiple-blocksize tapes are not updatable.

* Format wastes space with lots of null-padding

* No information as to when the tape was written.

* If you extract less than the full tape, files with multiple links will either not be extracted at all or will be linked to existing files in other subtrees. This is because tar stores the data for a multiply-linked file only once, with the first pathname it encounters for the file, and subsequent pathnames for the same file merely point back to the first pathname so they can be linked at extract time.

* A large file which is only partially allocated is written out to tape as if it were fully allocated, taking up tape space for zeros, and worse, forcing the file to be fully allocated when extracted.

I have worked out a new format which solves these problems. In addition, it stores all inode information about each file, and provides room for unlimited as-yet-unknown information about each file to be stored as well. The proposed format would also be better-suited than tar for other uses, such as floppy disk archiving and file transfer via data links.

For a tape format to be really useful, it must be widely available and widely adopted. The former is relatively easily arranged, but the latter will depend on user response. I haven't written any code yet, just a manual page describing the proposed tape format. If the user community can come to a consensus on the format, and, for that matter on the very notion that a new tape format should exist, then may be it will. I may even write it myself, but I'd rather someone else did it, especially someone at the Labs, where it would have the force of Law.

For lack of a better name, I have tentatively named the format 'FS' for 'File Stream'. Please read the following Section 5 manual page for the tape format. I am soliciting pro and con comments on it.

NAME

fs - file storage stream format

DESCRIPTION

The file storage command _fs_ is used to combine several files into one, especially to tape.

A file produced by _fs_ is in one or more 'segments' and contains information about each segment and each file stored as well the data for each file.

The physical blocksize written to magnetic tapes can be specified, but has no influence on the organization of the data on the tape, with the exception that the last physical block of a segment is padded to the end with nulls. New segments always start with a new 512-byte physical block.

There are three kinds of 'records' that appear on an _fs_ tape, one for segment information, one for file information, and one for file data. The first two of these consist entirely of an 'information block', which is a number of variable-length fields of human-readable ASCII data separated by newlines. The third type consists of an information block followed by file data. Records are of variable length, and the information block at the beginning of each record starts with a field containing the number of bytes in the record. Numbers are in octal.

Each segment starts with a record consisting of an info block containing general information about the segment:

* The number of bytes in the record.

* The number 'a' which identifies this as a segment info record.

* The string "fs".

* The number of bytes in the info block. A checksum of the first four items in this record.

* The _fs_ version number used to create the segment.

* The physical blocksize in bytes of succeeding blocks for the benefit of raw devices who care, such as magtape.

* A segment sequence number.

* The name of the installation.

* A checksum of the bytes in the segment info record.

* Optional additional data, checksummed but otherwise ignored.

Immediately following a segment record is the first file info record of the segment. There is one file info record for each file on the tape. A multiply-linked file is written to tape only once, with multiple filenames in its file info record. A file info block contains:

The number of bytes in the record.
The number '2' which identifies this as a file info record.
The number of bytes in the info block.
A checksum of the first three items in this record.
The number of filenames for the file.
The filename(s).
Unix file mode (all 16 bits, Version 7 Unix interpretation).
The number of links to the file.
User id name.
User id number.
Group id name.
Group id number.
Time of last access of file contents.
Time of last modification of file contents.
Time of last modification of file inode.
File size (see position of end-of-file.)
File reserved size. (Null. Not implemented.)
The device number.
The inode number.
The major device number.
The minor device number.
A checksum of the bytes in the file info record.
Optional additional data, checksummed but otherwise ignored.

Following a file info record are 0 or more file data records. File data may be written into more than one file data record to preserve the sparse-ness of a file containing unallocated blocks. At the beginning of a file data record is the following info block:

The number of bytes in the record.
The number '3' which identifies this as a file data record.
-The number of bytes in the info block.
A checksum of the first three items in this record.
The file offset of the bytes of data to follow.
The number of bytes of data described by this record.
The number of bytes of data in the record. If less than the number of bytes described by this record, then a section of the file containing zeros is implied.
A checksum of the bytes in the file data info block.
Optional additional data, checksummed but otherwise ignored.

The remainder of the record consists of the file data followed possibly by padding.

If a field in an info block is null, i.e. consists of only a newline, that is taken to mean that the information is not available. If particular, if the field should contain a number, the fact that it is null does not mean that the value is 0, but that the number is not specified. This can be fatal in many cases. In other cases, such as inode number, it is not. If a checksum field is null, then the field is accepted as is.

Checksums are 15-bit unsigned sums of bytes, where the bytes are treated as unsigned 8-bit quantities. A checksum does not count itself or its trailing newline. Times are in seconds since 00:00:00 GT Jan. 1, 1970 (unix standard time).

SEE ALSO
     fs(1)

AUTHOR
     Dave Yost, Rand Corp.

BUGS

News About C

A Portable C Compiler for the 32 bit World

Chris Terman of the Laboratory for Computer Science at Massachusetts Institute of Technology has developed a version of Portable C for the Motorola 68000, Intel 8086, and Zilog Z8000. Using the Motorola 68000 compiler, assembler and loader he has produced a UNIX for a Motorola 68000 development system. This Portable C compiler package operates in a full 32 bit environment.

The package will be available soon on a Usenix Association distribution tape. Western Electric licensees for 7th edition UNIX and UNIX /32V will be able to obtain this material. Several vendors are expressing interest in supporting this package.

C COMPILER(S) USAGE

The following statement concerning C compiler usage by UNIX licensees has been supplied by the Western Electric Patent License Office in Greensboro, North Carolina.

> Effective 8-15-80 Western Electric Company, Inc. has authorized the following C Compiler Usage between licensees.
>
> If your corporation or educational institution is licensed for at least two of our UNIX* Systems and/or Phototypesetter Systems, each of these systems includes a different C Compiler.
>
> We are aware that it might be convenient for you to standardize on one C Compiler so that your programs can all be written the same way regardless of the system to be used. Accordingly, you may use any C Compiler with any system you are licensed for. If you are also licensed to furnish object code for such systems to your customers, you may include any C Compiler in the object code you furnish. Of course, it will be necessary for you to modify a C Compiler from one system for it to operate with another. We are unable to assist or give advice with respect to any such modifications.

* Unix is a trademark of Bell Laboratories.

USENIX Conference - Winter 1981

Disclaimer

This document was prepared by Bill Croft of SRI, Mike O'Dell of LBL, and Clem Cole of Tektronix Inc. We believe the content is correct, but due to the vast amount of data that flowed past us as scribes in the four short days of the conference, mistakes may have occurred. We apologize to the authors if we have misrepresented anything that they have said. If you, the readers, wish any more information on any of these subjects, please contact the speakers directly. To help you in this task, we have included the speaker's affiliation and his address.

In most cases we have taken the author's abstract, as submitted to USENIX. After these we have added our own comments to help clarify and expand on some of the information contained in the abstract. These comments came from our notes.

Unless otherwise noted when we use the term V6, we mean UNIX Version 6 as distributed by Western Electric to the world outside of the Bell System. Similarly, V7 means UNIX Version 7, and PWB means PWB/UNIX Version 1.0 built on top of V6.

Opening Remarks

Tom Ferrin
USENIX Association

Tom was the host of the 1981 Winter USENIX Conference. After greeting us, he informed us that this was the largest USENIX Conference ever with over 800 attendees. With very little else we went right on into the main order of business.

Evolution of the UNIX Timesharing System

Dennis M. Ritchie
Bell Labs 2C517
Murray Hill, NJ 07974
(201) 582 3770

This paper presents a brief history of the early development of the Unix operating system. It concentrates on the evolution of the file system, the process-control mechanism, and the idea of pipelined commands.

The era covered in the talk was from the early days of UNIX on the PDP-7 to the initial porting to the PDP-11 and stops at the point when the 11 version was rewritten in C.

UNIX had its origins in CTSS and MULTICS. Some of the people involved at this early stage were Thompson, Ritchie and Ossanna. In 67 BTL left the MULTICS project, leaving Dennis and his group with no friendly machine. Money was eventually obtained for a PDP-11, but in the meantime Ken had found an old dusty PDP-7 off in the corner which used to be a graphics terminal. Ken's first program for the hardware was a space travel game which used his software FP package. At this stage cross assemblies were done on the 635 and downloaded via paper tape.

In 69 Ken and Dennis had the file system structure designed. The structure was pretty much the same with a super-block, I-list, and special files all existing at this time. The system calls read, write, open, close were all the same but used words for size fields. All tty IO was RAW, the individual programs (sh, ed, etc.) did their own ERASE and KILL processing. The file structure was not yet a true tree but had instead a general graph structure. Global path names with embedded slashes had not been invented yet; instead one had to use the link (ln) command to set up a relative path to the desired file. Mounting of file systems did not exist and thus the system was hard to reconfigure.

The process control primitives at this time were indeed primitive. There was only enough memory for one process at a time to fit in core. No switching could occur during IO. There were two processes on that system, one for each physical terminal. There was no exec or fork, and exit was somewhat different. The main loop in the primitive shell at that time was: close all files (to give a clean set of file descriptors to the following steps); reopen the terminal and read the command; overlay self with the command and execute it; eventually the command would exit which would read in a fresh shell. This scheme lasted less than a year, at which time the final form of the exec, exit, fork and wait calls was arrived at. Fork is not combined with exec because it was an add-on introduced in about a day of work and 27 lines of assembler by Ken.

The simple shell syntax for IO redirection and pipes was a significant change from the MULTICS I/O-switch concept. Pipe syntax evolved from a binary operator (infile sort outfile), to an extension of redirection (sort infile >pr>lpr>outfile). Arguments in this scheme were awkward (sort infile >"pr -2">lpr). In a fit of inspiration Ken finally put in "|" syntax right before he was scheduled to give a talk on the old scheme. McIlroy was a primary force in pushing for an easy pipe syntax.

There were also parallel compiler developments during this period. TMG was the first compiler on the PDP-7. Ken tried writing a Fortran compiler but gave up after three days. The programming language "B" was invented about this time. It produced compact interpretive code but was slow. B was also available after the 11 porting but ran into many type problems with byte addressing; C was thus born. When Dennis added structures, Ken recoded the kernel in C.

Someone in the audience asked about the origin of the name, and Dennis remarked that sometime in the PDP-7 days, Brian Kernighan had coined the pun on MULTICS.

The entire text of this paper is contained in lecture note 79, Language Design and Programming Methodology edited by Jeffery M. Tobias, from the CS lecture note series published by Springer Verlag, New York. Dennis also noted that work has started on the 2nd edition of the C Reference Manual (no major changes).

What's Happening at Western Electric

Larry Isley
Western Electric
PO Box 25000
Greensboro, NC 27420

Larry works for Otis Wilson in the Patent Licensing Division; Otis replaced Al Arms about a year ago. Larry had no new products to announce. He did restate the current offerings. A letter was recently sent out to all licensees clarifying compiler version problems: if you are licensed for more than one type of UNIX (e.g. V6 and photo7), you may run the latest of those C compilers (e.g. photo7) on all of your CPU's with no problems. You cannot however take a compiler you are not licensed for (e.g. a true V7) from a "friends" site and run it legally. <Ed. Note: See page 8 for the official statement -- WW> There was muffled booing from the audience when Larry announced that Western Electric was being more vigorous in their "site auditing" this year. Presumably this means Western Electric sends out a phone company repairman to look under all your rugs for improper software.

Current numbers of licensed UNIX sites:

commercial	176
government	96
educational	608
educ/administrative	17
total	897

Numbers of installations (licensed machines):

commercial	287
government	197
educational	1524
educ/administrative	51
total	2059

There are 200 to 300 installations within the Bell System.

There have been many requests in the past from people outside Bell asking of the availability of new packages (e.g. circuit design, UNIX 3.0, etc.). A committee has now been established with representatives from BTL, WE, and ATT, that meets monthly and comes to a yes or no decision regarding release of the packages in question. The first group of eight packages (Larry would not say which) has been submitted to the committee and answers on these should be available shortly. Each month in the future the committee should have further announcements.

What's Happening at DEC

Bill Munson
DEC, Continental Blvd.
MK1-1/D29
Merrimack, NH 03054
(603) 884 7436

Bill's group is there to enlighten DEC about UNIX and its opportunities. Among the sections in the UNIX group are engineering liaison, system engineering, and application engineering. Recently the group sent a memo to Gordon Bell explaining why DEC should "layoff" UNIX and not try to take it over and turn it into something else.

Bill's group offers a distribution tape, called "The DEC tape" (not to be confused with the device called DECTAPE) with device drivers for many of the new devices DEC is offering, such as RK07 and TS11. In March the tape distribution channels will change and they may start charging for it ($1500 commercial, $500 educational). They are working on things besides device drivers: C compilers and shells for other DEC operating systems.

You can help your local field service office better understand UNIX error diagnostics if you request that they contact Munson's group inside DEC and take a special DEC internal training course. Some of DEC's advertising literature now mentions UNIX instead of RSX/RSTS/IAS as a selling point; an example was the new ADML11 "CPU supercharger."

The New ARPA VAX UNIX Project

Robert S. Fabry
UC Berkeley, Computer Science Div.
Berkeley, CA 94720
(415) 642 2714

A new project in the EECS Department is being funded by the Advanced Research Projects Agency (ARPA) to provide a standard version of the UNIX operating system on the VAX computer for use by various ARPA contractors. The project is under the direction of Computer Science Professor Bob Fabry.

Traditionally, ARPA contractors have chosen a common hardware and operating system base to allow them to build effectively on each other's work. The first ARPA system was a time-sharing system developed by Project Genie at Berkeley for the SDS 940 computer. The second ARPA system was the TENEX system developed at Bolt, Beranek and Newman for the PDP-10 computer.

In the summer of 1979, various ARPA contractors chose the UNIX operating system for the VAX computer for future use. The major factors in this choice were that UNIX was portable in that it had been made to run on several computers with quite different architectures and that UNIX seemed to be easy to modify. In spite of its popularity, the ARPA contractors were able to identify a number of areas in which UNIX would have to be enhanced in order to be effective for the applications they had in mind. In the spring of 1980, ARPA selected Berkeley to provide the needed enhancements.

A major factor in Berkeley's selection was the excellent reputation of the current Berkeley VAX UNIX system, the first version of UNIX to take advantage of the paging mechanism of the VAX to facilitate programs requiring a very large memory. The paging facilities were developed by students Bill Joy and Ozalp Babaoglu under the direction of Computer Science Professor Domenico Ferarri.

The system which will be developed will be an enhancement of the current Berkeley VAX UNIX system. File access will be improved by allowing files to be logically coupled into the address space of a process. Pages of such files will be brought into memory only as they are used. If several processes are using the same page of a file, there will be only one copy of the page in memory; changes made by one process will be available immediately to the other processes.

A major effort will be mounted to improve the UNIX interprocess communication mechanism. The current ''pipe'' mechanism is considered inadequate. The mechanism to be implemented will facilitate message-oriented applications while still being compatible with the current stream-oriented programs. Processes will be able to communicate without taking into account whether they are on the same system or are communicating over a network. Additional networking options will also be provided. Local high speed networks as well as new ARPANET software being developed at Bolt, Beranek, and Newman, Inc. will be incorporated.

Because applications expected for the system include VLSI circuit design, image processing, large LISP programs, and so on, a number of performance enhancements will be made to improve the response time for large applications.

Bill Joy is playing a central role in the project which is budgeted for about two thirds of a million dollars over the initial period of eighteen months. This amount includes the cost of a medium size VAX system which will be used for the development work. In addition to staff members and faculty, the project supports a number of graduate students.

Berkeley is also seeking contributions of useful software for inclusion on future BSD's (Berkeley Software Distribution). There were initial hopes that much of the new software developed could be ported back to the PDP11; however this has proven to be impractical because of address space limitations. The original 2nd BSD (PDP11 version) is still being kept as current as possible and a new 2BSD distribution will be available shortly; the new VM/UNIX products which still fit on an 11 are included. They have started working with some new experimental Ungermann Bass local network hardware and hope to have a release available by the end of 81.

For information on 4BSD and 2BSD write to:

Laura Taung
Computer System Group of
Computer Science Division
UC Berkeley
Berkeley, CA 94720
(415) 642 - 7780

USENIX Business Meeting

USENIX Association
Box 8
The Rockefeller University
1230 York Avenue
New York, NY 10021
(212) 360 1182

Recruiting is still prohibited at the conferences. Enclosed with the next dues notice, the installations (but not individual members) will be voting on this issue.

Mel Ferentz announced that the 1980-1st distribution tape has been shipped to those that requested it; 158 tapes have been mailed so far. Mel has a list of institutions holding the tape. Any software contributed to USENIX distribution must have an accompanying release form showing what type of Western Electric license is required for each item on the tape; call the USENIX office for this form.

1981 dues will remain the same: $300 non-educational, $100 educational, $12 individual. Two to three tape distributions are planned for 1981. Harvard University is continuing to act as a wholesale distributor of manuals; currently in stock are V6, V7, PWB 1.0, and the Berkeley 4BSD.

Wally Wedel of the University of Texas is the editor of the USENIX newsletter, ";login:". In 1980 there were 5 issues (the Dec. issue goes out with the next 81 issue). This year Wally has committed for 10 issues, Jan-Jun and Sep-Dec. Contributions for the newsletter are cordially invited, by electronic mail if possible. ARPA address: login@utexas-11; UUCP: ucbvax!login@utexas-11. A dial-in is available, but not direct UUCP; the number is 512 474 5511, login to the user id login using the password USENIX to submit your contribution.

The June USENIX meeting will be held 24-26 June 81 (Tools on the 23), at the J C Thompson Conference Center, University of Texas, Austin. Wally Wedel is conference chairman. Space for vendor exhibitions and hospitality suites will be available.

Since the meetings are getting so large (800 at this one), they must be planned about a year in advance. John Donnelly is the Board member in charge of finding sites for future meetings and solicits recommendations. The Jan 82 site will be Santa Monica CA; Jun 82 is slated to be in the Boston area.

You can contact John at:
John Donnelly
NCAR
P.O. BOX 3000
Boulder, CO 80307
(303) 494 5151

The C/70 Micromachine -- A Hardware Overview

Alan G. Nemeth
Bolt Beranek and Newman
10 Moulton St.
Cambridge, MA 22138
(617) 491 1850

The BBN Computer Company C/70 computer system is based upon a general-purpose microprogrammable processor. This processor is designed to be easily microprogrammed and features a 32-bit, vertically oriented microinstruction. The processor has a 135 nanosecond instruction time, 20-bit data paths, and 1024 hardware registers. A significant portion of the processor bandwidth is budgeted for I/O processing to reduce the need for expensive peripheral controllers. Furthermore, the micromachine is well suited to the emulation of other computer architectures in that it possesses a large writable microcode memory and provides interfaces for special-purpose hardware to assist in macroinstruction decoding and memory mapping.

The hardware designed for the efficient execution of C and UNIX included a C instruction decoder and a memory address mapper. The instruction decoder computes a perfect hashing function of the C macrocode instruction set, thereby permitting fast dispatching to the appropriate emulation routines for addressing modes and op codes. The memory mapper provides a mechanism for mapping process virtual address spaces into physical memory space. The map contains 1024 mapping registers, allowing up to 8 process contexts to be contained in the map at once.

The microprogramming effort to provide an environment for C required approximately 9 man-months of effort and occupies about 6K of microcode memory. This code now occupies approximately 3K for the macrocode emulation, 1/2K for microcalls, and 2-1/2K for I/O microcode.

The special hardware instruction decoder and mapper for C resides on a daughter board of the main CPU module. The MBB architecture provides for other daughter boards for different macro instruction sets; e.g. the C/30, which is used in the ARPANET IMPs has a board which helps in the emulation of Honeywell 316 instructions.

There are currently 15 to 20 C/70's in existence now, by June that number will be 75 to 100.

The C/70 Macrocode Architecture

Carl D. Howe
BBN
(617) 497 3642

The BBN C/70 computer implements an instruction set specifically designed for the execution of the language C. It contains addressing modes for all of the major data types of C, as well as structure and array elements. The op code set is basically isomorphic to the operations allowed in C, thereby simplifying the translation process.

Of particular interest is the subroutine/function calling mechanism. The micromachine's 1024 hardware registers are used as a register cache to reduce the amount of time a program need spend saving and restoring registers. Also provided is the ability to call microcode routines using the same call instruction that is used to call macrocode routines.

On the C/70, I/O is handled in a uniform manner. All devices are controlled by means of common microcalls whose arguments are Device Control Blocks (DCBs) and I/O Control Blocks (IOCBs). Data transfer is performed by the microcode based upon the information supplied by the macrocode in these control blocks. Then error recovery and completion processing are performed by the macrocode. This division of labor between microcode and macrocode permits data to be transferred quickly by microcode while allowing the macrocode to control the transfer process.

The statistics gathered on the C/70 architecture indicate a saving of an average of 20% in code space over the PDP-11. The UNIX kernel for the C/70 is approximately 48K bytes and contains an ARPANET NCP implementation. The speed of the system depends upon the application being run; however, benchmarks show the system to be faster than the 11/70 in control flow, and slower in straight computation. Since most non-trivial programs require significant control structures, these programs tend to run faster on the C/70. This supposition is supported by a number of examples (e.g., nroff, etc.).

Porting UNIX to the C/70

Alan G. Nemeth
BBN

The UNIX Version 7 environment has been transported from the PDP-11 environment to the C/70. This required transportation of both the kernel of the operating system and a large body of user programs.

The effort began with the kernel of the operating system. First, a careful reading indicated areas of obvious focus of a transportation effort. These included memory management, process state saving, trap handling, and system call mechanism. The memory management unit was designed to support processes with large regions which are contiguous in both physical and virtual memory, and to support 8 simultaneous maps to reduce process switching time. The process state saving mechanism interacted strongly with the handling of register area overflow regions, and a series of microcalls were designed to allow the operating system to record a specific stack depth and return to that point. Trap handling and system call mechanisms were designed to model C function calls with enough information on the stack to permit resumption of previous activities.

The kernel was debugged before the disk controller was operational by allocating a region of memory to function as a small disk. Successive initialization programs first tried various system calls until they worked. Then a copy of the Bourne shell was loaded into the "disk" area, and the initialization code set to execute it. After some effort, this also succeeded in providing an extensive test of the kernel. At this point, the disk controller became functional and was used with a disk image with a more extensive set of programs which mostly functioned within a few days of the first trials.

The bulk of the system source code was then transferred to the C/70. The compiler and the 'make' command were viewed as critical first steps and came to life fairly quickly, although some problems developed in porting the compiler due to arithmetic differences. After this, most programs were merely compiled and then executed correctly. Modifications in the application programs were focused on changes made in the environment (8 bit bytes vs. 10 bit bytes, disk addresses of 20 bits vs. 24 bits, chars are unsigned, character order is right-to-left rather than left-to-right), with a few portability bugs detected.

LOCUS: a High Reliability, Network Transparent, Distributed System

Bruce Walker: Popek: Chow: Edwards: Kline: Thiel
University of California at Los Angeles
3804 Boelter Hall
Los Angeles, CA 90024
(213) 825 7307

A high performance, high reliability distributed operating system has been developed at UCLA. It supports transparent access to data whether the files are foreign or local and shows performance comparable to standard Unix. File names are independent of file location. Flexible, automatic replication of storage is supported in the design. Graceful partitioned operation is provided.

The overall system architecture will be described, as well as experiences in achieving good performance. The rationale for network wide demand paging, as well as the network wide naming strategy, will be presented. The specialized protocols developed for the local network hardware will also be discussed. Performance data will be given.

Preprints of a paper to be presented in the next ACM SIGOPS SOSP are available from Bruce Walker. The system only requires kernel modifications, no user-level code is affected. The operating system is "derived" from V7, more was put in than removed.

The file system has been reorganized to allow network-wide file naming. Special device numbers are global file system numbers. One affect of this is that all pathnames, even the /dev and /etc names, must be unique throughout the network. A file replication mechanism has been added; the superblock assigns a range of inodes to each site. Copies of the same file may exist on multiple machines; an elaborate synchronization and locking mechanism ensures that all copies are consistent.

Performance measurements show little if any added overhead compared to standard V7; the elapsed time degradation for a block accessed over the net versus one present locally is 9 ms. per block on reads. They are using a one megabit ring network.

C on the FPS-164

K. Wilson
AP Support Group
Cornell University
Ithaca, NY 14850
(607) 256 5169

Ken is working on bringing up C and F77 for his Floating Point Systems array processors. They have been using the AP190L (same architecture as the 120B) and have a F4 compiler and microcode assembler for it on their IBM 370/168. They would like to talk with anyone else who is interested. They will soon have the new FPS AP164 with 1.5 Megawords of 64 bit memory and are especially eager to bring up better software tools to use it. The 164 will first be interfaced to a VAX, and later to their IBM.

A Truly Portable IO Library

Daniel R. Strick
Univ. of Pittsburgh
833 LIS Building
Pittsburgh, PA 15260
(412) 624 5218

The stdio library distributed as part of UNIX V7 and 32V provides only the most basic IO buffering facilities. Unusual requirements are sometimes satisfied by adding a few bells and whistles to this library. More difficult IO problems typically result in ad hoc solutions: a collection of special purpose IO routines is used for a single application. When a application at the Univ of Pittsburgh developed a few exotic requirements (e.g. unrestricted random RW access, simultaneous access to a single file through different IO streams, a disk block cache in user memory, more rapid access to full word data items), the parts of the library that involved buffering were completely rewritten. Since the new features were exotic, they found few additional applications until activities on an independent RSX system using a standard IO package were required. The RSX system supported a C compiler (Whitesmith's), but no other tools. An IO interface similar to stdio was desired, but the UNIX library could not be ported.

Daniel's software has been submitted to USENIX for inclusion on a future distribution tape. An audience member pointed out the the stdio package provided with the Berkeley 4BSD has similar features.

Using UNIX to Develop an IBM front-end for Students

Peter Hardie
Univ of Saskatchewan
Rm 65 Arts, Academic Computing Services
Saskatoon, Sask, Canada S7M 0W0
(306) 343 2638

DEUS (Data Entry at the Univ of Sask) is a UNIX based program and data preparation system with RJE access to an IBM host. The system currently runs on a PDP 11/70 with 1/2 MB of memory and DQ11EA synchronous interface. DEUS supports up to 40 student users, concurrent with 4 data entry operators, an OPSCAN-17 mark sense reader and 3 ports for programmers and instructors. The total student population for the fall of 80 was 1060.

Students use a modified UNIX editor to enter their program and data files and to submit the job to an IBM 370/158 using a HASP multileaving driver for the DQ11 interface (9600 baud). The resulting listing is returned to the student's directory from which it can be examined on a terminal or printed on the line printer.

The data-entry operators use a separate software package (also locally developed) called DESK (Data Entry Sans Keypunches) which permits entry and verification of data using definable screen formats for display of the data. This subsystem is used by full-time data entry personnel for entering both student work and general campus data.

A timed logon feature was required to restrict student terminal sessions to less than 30 minutes when there are long queues for terminal access. The students stay in a modified editor all the time which has certain escape commands: !send - sends a job consisting of program and data; !notify - tells the student when his job has come back; !take - allows inclusion of instructor generated material in the students job; !print - print a listing; etc.

The DQ device driver handles the low level protocol work such as IO errors, timeouts, and line stats. User level processes (console, sendperm, receive, and give) interact with each other and the driver via semaphores to carry out the RJE functions. <Since this is all newly written software, it was unclear to me whether they had considered running the RJE software supplied with PWB/UNIX 1.0 -- Scribe>

DEAFnet

Kenneth Harrenstien
SRI International
333 Ravenswood Ave
Menlo Park, CA 94025
(415) 859 2236

Deafnet is part of a DHEW/OSE-sponsored project being carried out by SRI International to study telecommunication services for the deaf. A demonstration facility consisting of an 11/70 running UNIX is being used to provide electronic mail services to the deaf community in Washington, D.C. SRI and the government are using the experiences with this system as guides in developing a nationwide network for the deaf. Some unusual aspects of the system include: (1) the use of special modems and software to handle the Baudot TDDs which many deaf people currently use to communicate over the telephone, (2) the need for a very simple user interface to the general public, and (3) the use of MMDF for mail transport to other sites, including the ARPANET via gateway.

The system connects an 11/70 UNIX at Gallaudet College through an MMDF phone-link to an 11/40 UNIX at SRI and from there to a PDP10 TENEX at DCC via the ARPANET. MMDF is similar to UUCP, but provides additional gateway functions.

UNIX was chosen for several reasons. The PDP11 family offered a good series to build a nation-wide net upon; the UNIX OS was found to have the flexibility and capabilities needed; certain network software such as FTP and MAIL services were available. The project required total control of all software, even in the kernel, since things such as the tty driver had to be modified.

They encountered numerous problems. The user population is computer naive, has a language adjustment problem (English is a 2nd language for most deaf people), and needs special services that normal systems don't provide. Terminals were the largest problem; most deaf people own surplus 5-level Baudot uppercase-only terminals (which are horribly slow -5 chars per second) and use inexpensive, non-standard Weitbrecht modems. To make use bearable, extraneous output had to be removed and escape mechanisms devised to substitute for lack of lower case or control keys.

Response to the system in the areas serviced has been enthusiastic; over 300 users are in the DC area. The biggest complaint is that there are still not enough other people to talk to. The prospects for the future look promising. The California Public Utilities Commission has recently ruled favorably by mandating that publically provided free terminals and modems for the deaf will follow the ASCII and Bell 103 300 baud standards; this will allow future systems to be built more easily and provide greater communication between deaf and hearing people. A series of 4 reports on the DEAFnet project is available from SRI.

ANGUS = MASCOT + PWB/UNIX 1.0 + TIPs + MENU

David Tilbrook
Exploratory Development Group
Bell Northern Research / Toronto
522 University Ave
Toronto, Canada
(416) 589 0196

EDG is involved in research relating to system design, prototyping, testing, maintenance, and portability. The subject of this talk is the integration of systems and disciplines that are being developed and explored. The major components of the systems are:

PWB/UNIX 1.0: The merits and use of this system for software development needs no expansion.

MASCOT: The Modular Approach to Software Construction, Operation and Test. An approach to design for multiprocessor real time systems. Part of MASCOT is a set of kernel primitives that support its approach to system design. The BNSR UNIX/MASCOT system is just one of many MASCOT systems but to the best of my knowledge the only one in North America and the only UNIX implementation thus far. A presentation of

MASCOT was done at the Delaware USENIX meeting.

TIPs: The Tilbrook Information Processing System. A set of functions that support information retrieval and manipulation. This system, which is available through USENIX, is used extensively within BNSR for a variety of information needs. TIPs was presented to the Boulder USENIX conference.

TIPs/MASCOT data base servers. Over the last six months a number of TIPs/MASCOT data base modules have been developed. Such tools are now being used extensively both for production systems and system prototyping.

The MENU system: Yet another integration of TIPs, MASCOT, and UNIX is the Menu programming language. Menu source files are in TIPs format and retrieval and large part of the interpretation is provided by the standard TIPs functions. It provides interfaces to the UNIX shell and file systems and can send and receive messages via MASCOT IDA's. The major use of Menu at BNSR is as an interface to the UNIX shell for novice or casual users (it is the default shell for a large number of users). However, its uses range from babysitting (for weekend visitors of all ages), sophisticated system support tools (system backups, new user procedures, etc.), computer assisted learning courses, games, telephone answering systems, and the Cocopuffs joke book.

USGS/UCB V7 UNIX System

Bill Jolitz
USGS
Office of Earthquake Studies
345 Middlefield Ave
Menlo Park, CA 94025
(415) 329 8057 x2975

Bill reviewed some of the kernel improvements available from the Berkeley and USGS systems: 1K block file system (the single biggest throughput improver). External buffer pool of 60 to 100 buffers. Larger, external clist. hashed I-list, and a hashed buffer table. Some simple TENEX like load average statistics. Text overlays; the same scheme works in both user and kernel space. Additional tty discipline that can be turned on to make terminal act like a TENEX tty. "Stacking" tty line disciplines. Misc bug fixes.

Administrative features include: Dynamic file space quotas. Special protection for /tmp file to prevent removal of editor tmp's (unlink only allowed if you are the owner). Expanded accounting. Performance monitoring. Privileged accounts (a level below superuser). All of these features are MACRO'ed into the source so that a only a setsub of the features need to be turned on at any installation.

Ongoing efforts include: Sturdy file system mods <see later talk by Joy>. Device drivers that handle ECC correction on raw files. The "job control" package from J. E. Kulp that was described at the Boulder meeting.

The text overlay system developed at Berkeley was briefly discussed with its limitations. Not all I/D programs will work D only; EX/VI will fit, but F77 does not (without a lot of hacking). Thus the overlay problem only helps large TEXT small DATA programs.

Compact and Simple Kernel Overlays

Kenneth Harrenstien
SRI International
333 Ravenswood Ave
Menlo Park, CA 94025
(415) 859 2236

I have implemented a "kernel overlay" scheme which essentially allows the UNIX kernel to contain any reasonable amount of code. No changes are required to the C code and about 50 instructions are added to the machine support code, half of which is setup that a boot loader could perform. Speed appears to be little affected. Almost any kernel modules can be selected for overlaying -- it is very flexible. This feature is achieved through modifications to the "LD" loader, and the method is usable on either 40 or 45-type machines, although it is of course most urgently needed for machines without split-I/D space. In particular I am using it to run an ARPANET NCP on an 11/40; another use would be running V7. (I am also using user-space kernel buffers (UCBUFMODS), so that data space is not too much of a problem either).

This scheme is similar to the one that follows and the UNSW technique. It is an idea that seems to have originated at Calgary under V6.

Version 7 Conversion Tools

Michael D. Tilson
Human Computing Resources Corporation
10 St. Mary Street
Toronto, Ontario, Canada M4Y 1P9
(416) 922 1937

This talk describes solutions to several problems involved with converting to UNIX version 7.

The first problem involves the use of Version 7 on smaller PDP11's, particularly the PDP 11/23. A memory-management scheme was developed to allow the kernel code to fit in non-I/D space. This scheme resembles one developed by the University of New South Wales, but it is much simpler to use. Only a few lines of C source need be changed, along with a dozen or so lines of assembly support. A program similar to V6 "sysfix" does the rest of the work automatically. No additional input is required. This scheme has been found to be efficient both in space and time.

An extension of this scheme will allow I/D programs (e.g. F77, lint) to run on non I/D machines.

The second problem involves the conversion of large quantities of Version 6 code, some of which may exist in binary form only. A modification to UNIX was developed to provide Version 6 emulation under a V7 system. Programs may then be converted at leisure, or not converted at all, as desired.

Features of the overlay scheme are as follows. The ka5 mem management register (takes away 8Kbytes from the kernel) is used as a window into a library of upto 64Kbytes of additional text space. Usage of the mem management registers is then, ka0 = low.s, mch.s, main.c, other always resident functions, ka1-4 = data and bss area (never mapped), ka5 = text window, ka6 = _u struct, ka7 = dev page. There are no restrictions on calling order; only about 3 lines of C and 60 lines of assembler were changed. Adding a ka5 stack pointer to the _u area eliminated the modifications required by the UNSW scheme concerning argument counts. A program "23fix," similar to "sysfix," takes a separated I/D /unix a.out and relocation bits to produce a mapped /unix a.out. No attempt was made to place things such as tty.c and dh.c together on the same page, but this gained little in performance anyway. No loader mods were required.

The system performs well. The csv/cret overhead for a mapped call was only 2.5 times greater than normal. Six more bytes are added per mapped call. Typically runs with 12 to 30% of the functions mapped. A floating point simulator was added to the kernel.

Multi-controller disk driver

Peter Staubach
Univ of Oklahoma Engr Computer Net
202 W Boyd CEC107
Norman OK 73019
(405) 325 5370

Talk discusses a "universal" disk driver which attempts to deal with the RP04/05/06 disk family and the RM03/05 family, as implemented by several different controllers (DEC and alien) simultaneously. The goal is to allow a considerably mixed configuration with only one copy of a driver, with hooks for dealing with design misfeatures of various vendor's controllers. The major design issues will be discussed, such as automatic drive-type recognition, multiple unit scheduling in the face of emulated disk drives (e.g., 3 RP04's on one CDC 9766 drive, but 2 9766's in use), and other such issues.

OU has 1 rh70 and 2 sc70 controllers. The 8 bits of minor device were split up as: 2 bits - controller, 2 bits - drive, 4 bits - file system. Constants that used to be defines are now table driven: NSECT, NTRAC, SDIST, RDIST, GDIST. The scheme is not yet fully general; plans in the future are to add: auto drive type recognition and configuration based on the drive-type CSR; throughput maximization (overlapped seeks); cleanup after hardware problems such as power fail; bad blocks.

Floating Point Bug in PDP-11 UNIX

James A. Reeds
Statistics Department
University of California
Berkeley, CA 94720
(415) 642-2781

User programs which catch their own FP exceptions on an 11 will be betrayed by the kernel when an FP exception occurs when the CPU is in kernel mode. Then the SIGFPE is posted but not processed, and when the CPU returns to user mode the user process is not aware it incurred a SIGFPE. As a result Johns Hopkins Basic+ will, one time in about 3000, compute 10 - int(.5) as 20! This can be fixed by trap() jabbing the scheduler whenever a kernel mode FP exception occurs. This will force all pending signals to be processed before the user process resumes.

The problem exists on all V6 and V7 PDP11s. This fix includes the Amsterdam FP save bug fix as well. Another problem is that the FP error regs are read only, they cannot be saved and restored; therefore they should be read only by the kernel. Fortran and Basic were using the "stst" instruction and getting other users registers; they should instead make a kernel sys call to get this information. For more information see: *UNIX Problems With Floating-Point Processors* by Bob Campbell, Ed Gould, Vance Vaugh, Jim Reeds of UCB.

Real-Time Analog I/O on UNIX without System Modifications. The DEC Laboratory Peripheral Accelerator (LPA).

Asa Romberger
484 Cory Hall
University of California
Berkeley, Ca 94720
(415) 642-1035

The DEC LPA-11 is an analog I/O device that is microprogrammable. It consists of a microprocessor that controls communication with the UNIBUS, an interprocessor buffer, and a microprocessor that controls a second UNIBUS to which program controlled analog and digital I/O can be attached. The master microprocessor can be down-loaded from the main machine; the slave microprocessor code is in rom. The microcode as it comes from DEC allows for multi- buffering between the LPA and the main machine.

I have developed a UNIX driver that uses the device for speeds up to 50 KHz with users on the system, without any modifications to other parts of the operating system.

The talk will cover our choice of the device, the structure of the device, and the structure of the driver and supporting user programs that I have developed.

Terminal Linking

S. J. Leffler
Sytek Inc
1153 Bordeaux
Sunnyvale, Ca. 94086.

W. A. Shannon
DEC, Continental Blvd, MK1-1/D29
Merrimack, N. H. 03054.
(603) 864 5044
<both formerly with Case Western Reserve University>

A new line discipline has been created for release 7 of the UNIX operating system to support interactive communication between two terminals. This facility, called "terminal linking," was designed specifically for the following applications:

(1) Interactive dialogue between two users, a la write(1).
(2) Interactive assistance for novice users.
(3) Simple front-ending of computer systems using only RS-232 lines.

The ideas behind terminal linking originate from a similar service found on the TENEX operating system. The current implementation appears to be very efficient, though it suffers from some security flaws. This paper discusses the implementation of terminal linking and describes the user facilities which have been built atop it.

The technique used in the "cu" program has been used, quite reliably as a matter of fact, to communicate with both UNIX systems and non-UNIX systems under both versions 6 and 7 of UNIX. However, communication was always limited in speed, and normally introduced quite a bit of system overhead because the simple function of connecting the two terminal's input and output required many UNIX read and write calls. Consequently, we decided to provide a facility whereby two terminals could have their I/O queues "hooked" together within the kernel. This facility, eliminates most user process overhead in a situation such as described above, while providing many capabilities not possible with the normal approach.

In addition to the application just described, the ability to link the character queues of terminals together provides a nice service to use in implementing interactive communication between UNIX users on the same machine.

The software can be obtained from Bill Shannon at DEC by sending him a tape. A document is also available detailing their system. The software works well in it's current state, but there are some security issues that may be broached when using this scheme.

A Low-Cost Terminal Multiplexer for UNIX

Michael D. Tilson
Human Computing Resources Corporation
10 St. Mary Street
Toronto, Ontario, Canada M4Y 1P9
(416) 922 1937

This talk describes an expèrimental low-cost terminal multiplexor based on a Motorola 6800 microprocessor. The multiplexor connects to a UNIX machine via a single high-speed <9600 bps> serial line, and supports up to 8 terminals or printers.

The multiplexor is capable of off-loading some of the more costly UNIX terminal output functions. The multiplexor driver is designed to allow efficient use of the serial line, with CPU overhead comparable to that of an expensive DMA interface.

Mike would like to hear from others who are interested in this approach. The technique is particularly useful for remote TTY clusters and TTY port contention. The protocol used between tty.c and the 6800 has the following features. Data and control are intermixed in the same byte stream, the 8th bit is used as the control indicator. Data compression (run length encoding) is used for maximum bandwidth sharing. Little error correction is done since the line is inherently reliable, but error recovery (resynch, etc) is provided.

Control commands out from UNIX to the 6800 are: switch to line N; replicate the following byte N times; acknowledge (each 6800->UNIX data segment is acknowledged). Commands from 6800 to UNIX: switch to line N; halt line N; resume line N.

Unfortunately, this scheme has a major drawback. It assumes that you will not wish to send anything but 7 bit ASCII down the TTY line.

A 4 line mux is installed and working for <$1000; he uses Wintek 6800 boards. The 4 9600 users sharing one 9600 mux line are unaware of the sharing, but then again they do not use screen editors. In the future Mike would like to move more of tty.c out to the 6800 to reduce kernel overhead even further. He would like to define a single standard virtual terminal and the 6800 would then convert these codes to device specific ones.

A Multi-host Front End for Terminal Communications

Ronald L. Broersma and Robert A. Unger
Naval Ocean Systems Center Code 9121 (B)
San Diego, Cal. 92152
(714) 225-2148

This presentation describes the architecture and present status of a multi-host intelligent front end for terminal communications. Several UNIX systems at NOSC are connected to the system, which provides a microprocessor for each terminal, and solves the problem of providing communications from hundreds of terminals to a variety of computing systems. It allows many terminals to contend for ports, reduces overhead in the host systems, allows higher communications speeds and more ports through use of parallel channels, provides access to multiple hosts, and reduces the number of communications lines needed throughout the system. It provides a consistent terminal interface for each user and provides in-line editing capabilities over and above the normal character and line delete functions. Many of the mundane tasks such as expanding tabs, echo, gathering lines, character translations, buffering characters, providing flow control and delays, and the like can now all be moved to the front end, returning many precious CPU cycles back to the system. This can be done while maintaining all the UNIX functionality.

There is indeed one micro per user terminal but they still estimate that one could build this system in-house from their plans for $300 per terminal. They have built a UNIBUS card that interfaces to the system over a high speed parallel bus and looks like upto 120 DMA ttys. Stty/gtty functions are propagated to the micro controlling the terminal; in turn, at login time the micro passes back the tty type to getty.

A Crash-resistant UNIX file system.

Bill Joy
UC Berkeley, Computer Science Div.
Berkeley, CA 94720
(415) 642 7780

Credit for the various pieces involved goes to Ted Kowalski of CMU and BTL, Mike Accetta of CMU, and George Goble of Purdue Univ. Kowalski wrote the "fsck," file system check program, which does all that icheck/dcheck did and more and also repairs errors that it finds. In the Purdue V6 system Goble had added code to ensure that inodes and indirect blocks on disk were always in a consistent state so that the worst that could happen in a sudden halt or crash was that the file system may have some dups in free, but never dups between files. Accetta enhanced these and added a few of his own.

In the current Berkeley distribution (4BSD) all of these techniques are provided as well as some additions. A few more inconsistency conditions were discovered and corrected. There appears to be little performance degradation required to keep a running disk consistent. File deletes do take slightly longer. The fsck program runs much faster on the VAX compared to an 11 since the tmp file is no longer required (all that information will fit in VAX memory instead). A new -p option to fsck will check a number of file systems in parallel, which can save a lot of time if you have separate controllers. Fsck was also made a little smarter in that it can make "reasonable" a file. Thus fsck is now invoked automatically as part of the VM/UNIX autoreboot procedure if the system went down unexpectedly. If there are any serious errors, fsck waits for manual confirmation before attempting repair.

The conclusion is that UNIX no longer has a "fragile" file system that requires babysitting by a guru.

EUNICE: A system for porting UNIX programs to VAX/VMS

David L. Kashtan
Artificial Intelligence Center, SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025
(415) 859-5830

EUNICE is a software package for transporting Version 7 UNIX software to VAX/VMS. It provides a UNIX system call emulation package to allow UNIX programs to run under VMS with little or no modification. With EUNICE, users have a choice of operating in a UNIX environment using the UNIX shell or in a VMS environment but with the capability of using tools available on UNIX. File name coercion in EUNICE allows the user to enter file specifications in either UNIX or VMS forms. Data coercion in EUNICE allows UNIX programs to read and write both VMS files and UNIX files and to read VMS directory files as though they were UNIX directories.

Vax/UNIX Enhancements and Directions

Bill Joy
UC Berkeley, Computer Science Div.
Berkeley, CA 94720
(415) 642 7780

Some of this talk overlapped with Bob Fabry's above on the UCB ARPA VAX UNIX support effort.

In the file system area Bill is working towards higher performance through the use of locality and new algorithms. The current file system organization places the inodes and data at opposite ends of the disk pack; consequently the disk arm is bouncing around a great deal. Making this problem worse is the rapid increase in disk storage density making 300 and 600 megabyte file systems commonplace. A new organization called "cylinder groups" will greatly reduce this problem. Each cylinder group is sort of a microcosm of the original file system layout, with a section of inodes and a section of data blocks. Note that this is not the same as merely partitioning a physical disk up into separate minor device file systems as was done in V6. The new scheme will attempt to place data and its inode information in close proximity to each other. Even now, file systems tend to get organized randomly after a great deal of file creation and deletion, so in the new scheme a background process will shuffle files transparently between cylinder groups to achieve optimal access times. A new "spill" algorithm and performance measurement software built into the disk driver will allow fine tuning of the organization.

Several new IPC (interprocess communication) mechanisms are being investigated, including CMU's and mpx. Developments are underway in the local networking area using different hardware transport mechanisms such as Ethernet and Chaosnet. BBN is working on the VAX ARPANET interface, using the TCP/IP protocols.

A better way to interconnect tty line disciplines is being worked on. They currently have up to 7 loaded at a time, but at present they are all mutually exclusive. What is desired are disciplines that "stack" so that several can be in effect simultaneously for the same tty line. The low level tty drivers (such as dh.c) are being altered so that they are unaware of the tty struct. Better buffering methods (than clist) are being developed.

In the virtual memory area current efforts are to provide higher performance and to allow sharing. They have decided upon a segment based VM scheme with "copy on write" like TENEX.

Around March a 4BSD update will be available with: VAX/750 support, F77 fixes, bug fixes, multi-MBA support, and directly exec-able shell files (the first line of a shell file can now be "#!interpretername", to allow sh/csh coexistence).

The VAX-11/750: Comet Haley or Kohoutek?

Rob Pike
Bell Labs 2C521
Murray Hill NJ 07974
(201) 582 7854

The Computer Science Research Center at Bell Labs is acquiring several VAX-11/750 "Comet" CPUs, with one installed and operational and four more arriving in early 1981. This talk will discuss, from a UNIX point of view, the VAX-11/750 and how it compares with other DEC hardware on which UNIX runs, and report on the process of bootstrapping the first VAX/750 at Bell, and fitting it into the computing environment and network within the Computer Science Research Department. Aside from the usual teething pains associated with new machinery, some events in the bootstrap process provide insight into the potential success of the VAX/750 as a UNIX machine.

Rob estimates the VAX 750 is 60% of a VAX 780 for 40% of the price. The VAX 750 uses gate array technology. BTL currently has three of them and more on order. The RK07 and TS11 peripherals which DEC tries to bundle with the VAX 750 have numerous problems. In trying to compare a VAX 750 with other PDP 11's and the VAX 780 the following MIPS (million of instructions per second) figures were given:

VAX/750	1.18
VAX/780	2.39
11/70	3.21
11/45c	2.46
11/45	0.98
11/23	0.51
11/03	0.19

The VAX 11/750 (Comet) is a machine with the following characteristics:

- vax architecture
- 1 UNIBUS, no massbus (maybe spring), no FPA
- UNIBUS is different from VAX/780 (closer to 11 unibus)
    - similar to code for but interrupts much simpler (hardware does most of the work)
- physically much smaller than 780 (waist high, looks like a disk)
- has console and TU58 (DECtape II!)
- packaged from DEC with RK07 or RM80 (Winchester) (when massbus comes)
- tape is new TS11, a totally different drive to program (packet controller interface)
- speed for pure CPU is around 60% of VAX/780 (but varies widely with application); relative instruction timings compared to VAX/780 are quite different
    - see instruction timings:
        - RALU slow, cache fast
        - note lack of FPA
    - internals:
        - gate array technology
        - 80 bit microprogrammed

Rob's first Comet had 3 RK07's and an Emulex/Fujitsu disk subsystem for large storage. They bootstrapped via a VAX 780 under 32V. The system had some problems, some were in the microcode, some in the peripherals:

- probe<rw>:
    - if the first addr good, second bad, and the translation buffer is empty ==> you get fault anyway (!)
- movtuc:
    - if source and dest overlap, you get unpredictable results (core dumps, etc.)
    - instruction actually correct, but architecture book doesn't specify it well. The book will be fixed, not VAX/750

  - only Unix code that uses it is printf; we just rewrote it

- translation buffer empty:

  - (a rare event, but....) may cause problems (e.g. probe above)
  - suspicion of microcode loop if instruction in last word of page gets write protect violation (e.g. after a fork)

-

  fp bug: (2¬-1 + 2¬56)+(2¬-1 + 2¬31) gives sp result
- noisy power control

  - can screw up peripherals; DEC sends out the machines with a waiver

- bad RK's (more later)

The problems were largely dealt with, but required some "futzing." An example is that you really do not <u>need</u> the probe instruction. Once stable, the first Comet was put on the Research Network. At that time, they tried to bring up "pure Joy." In January, 4 new Comets arrived and they are working quite well. These machines do not have tapes, the dumps are produced through the network to a machine with a tape. The bootstrap process was done for the 1st Comet by putting a VAX 780 system on a RK07 pack and moving to the Comet. The later VAXEN were bootstrapped by copying disks. The major changes to Unix to convert from the VAX 11/780 to the VAX 11/750 were:

- MBA->UBA

  - haven't tried non-buffered DMA

- Unibus vectors interrupts into SCB, not by supplying vector address in register (more like 11 UNIBUS)
- VAX/780 ignores high bits in PCB register, the VAX/750 doesn't

Rob characterized the Comet as a Unix machine as "generally quite good." He mentioned that a massbus would help as disk throughput increased, but not to worry because a massbus should be available soon. He claims that as a production machine it seems to have been been shaken out at the design level (i.e. it's more consistent). The VAX/780 has had a history of problems that the users had to find. But Rob also warned that the Comet may be buggy. He noted that there is no need for the DZ11/KMC11 kludge. because a DH11 can be placed on this Unibus. As everyone had hoped, the user binaries from the VAX/780 run fine, and only 11 idef's need to be placed in the kernel to make it run on the Comet. There were 4 in locore.s, 2 machdep.c, 4 cons.c, and 1 in uba.c. The device drivers were the same as before, except that the 32V that was available to the world outside of BTL, does not have the Unibus DMA support. DEC wants to sell with the VAX 11/750 with RK07 and TS11; these are Rob's notes on these peripherals.

- RK07: troublesome

  - RK doing dma does bus request every 4 microsec, but only uses 2 microsec of it, so gives up the bus for 2
  - if there is a second dma device on the bus, the best they can do is alternate bus cycles. but if 2nd device takes two cycles per dma, RK is locked out for a long time, and when it gets bus back only uses 2 microsec again
  - RK has a 66 byte (!) buffer, and this overflows very fast if there is such a 2nd dma device (e.g. emulex disks) result: DLT errors, especially on Joy's system
  - if you get one of these during a write, however, and a factory ECO isn't installed right (and it wasn't on ours), the write pulse stays on and it zeroes the sector headers, requiring reformatting
  - servo system unstable; every few months get DROT errors on write. can't recover in software. must retune drive.

- TS11: whoa

  - hopelessly difficult to mount tapes
  - like the TE16, there is a small nylon ring designed to prevent skew errors, but frequently gets skewed itself and results in skew errors
  - generally flakey. one day both UNIX and VMS wrote to a tape without detecting an error even though tape didn't move
  - passage of time usually fixes one of these catatonic conditions

- Power Supplies

  - errors in sequencing of control signals sometimes cause bus POWER OK signal (DCLO.LI)to be true before power supply is up to 5 volts.
  - signal can even float or go indeterminate
  - confuses peripherals
  - DEC sells machine with waiver

Rob finished off with the following statement:

"It makes a comfortable personal computer."

<u>VAX News from DEC</u>

A P Stettner
DEC, Continental Blvd.
MK1-1/D29
Merrimack, NH 03054
(603) 884 5485

Armando is in the same group with Bill Munson. "The DEC tape" distribution tape that they have will boot and sysgen on all DEC PDP11 CPU's. It includes device drivers for all of DEC's latest oddball peripherals: RL, RN, RK07, TS11, etc. They also have a VAX version that boots on either a VAX/780 or VAX/750. They are working with Berkeley to incorporate these mods into 4BSD.

VAX-UNIX Networking Support Project

Robert F. Gurwitz
Bolt Beranek and Newman, Inc.
10 Moulton St.
Cambridge, MA 02238
(617) 497 3041

An ARPA-funded project is currently underway at BBN to provide networking support for the VAX-11/780, running UNIX. The overall purpose of this effort is to provide the capability for the VAX to communicate with other computers via packet-switched networks, such as the ARPANET. Specifically, the project centers around an implementation of the DoD standard host-host protocol, the Transmission Control Protocol (TCP). TCP allows reliable communication with ARPANET hosts, as well as hosts on networks outside the ARPANET, by its use of the DoD standard Internet Protocol (IP). The implementation being developed is designed for the VAX, running VM/UNIX, the modified version of UNIX 32/V developed at the University of California, Berkeley.

This presentation discussed the details of the implementation, including design goals, protocol dependent features, operating system dependent features, and the user interface. In addition, the organization of the network software was presented.

Goals of the implementation are to: maximize throughput, minimize queuing between levels and copying of buffers, minimize modifications to other kernel code, and to have all the low-level software resident in the kernel - rather than in "ncpdaemon" processes as was done in the ARPANET NCP implementation.

Testing is now underway; the first release to experimental test sites will be on March 15. The software is in the public domain and will become a part of the Berkeley 4BSD. A paper describing the implementation is available from the ARPANET NIC, number IEN2068.

A Network Peripheral

Steve Holmgren and John Mullen
Mitre Corp.
1820 Dolly Madison Blvd.
McLean VA 22102
(703) 827 6375

To date, local and long-haul computer networking has been expensive in terms of host machine memory requirements and host operating system requirements (e.g. inter-process communication, non-blocking I/O). The UNIX ARPA network NCP software is a case in point; the software requires roughly 14K bytes of precious kernel address space and barely fits in 11/40 systems. A rudimentary message-based IPC mechanism was developed internal to the NCP. Non-blocking I/O was avoided at the expense of a process per simplex communication path. These problems are exacerbated when older operating systems are involved.

Recent microprocessor developments (e.g. Zilog 8000, Motorola 68000) have provided the performance to support standalone versions of the NCP class of software. This affords the opportunity to support inexpensive, high-performance communications in a generalized way.

Offloading NCP-like software to an attached microprocessor peripheral introduces the requirement for residual host software which directs the operations of a network peripheral (e.g. open a connection, send data, etc).

This presentation discusses the C implementation of DoD standard Transmission Control and Internet protocols on a Z8000 base. After a short description of the system, experiences with system performance and UNIX residual software will be presented.

This software, like the BBN TCP software, is also in the public domain. TCP/IP currently resides in a Z8000 and is written in C. Between Z8000's over a 880Kbaud channel they are getting 600Kbaud (TCP) and 350Kbaud (IP) sustained throughputs. A "network access protocol" (a front end protocol) between a UNIX and the Z8000 has been designed and is currently being implemented. By June 81 the NAP will be complete and UNIX kernel code will exist. <It would be interesting to compare the NAP used above to the one used in RAND's front end below. -- Scribe>

The RAND Network Front End

Steve Tepper <greep@RAND-unix> and Mike Wahrman
The RAND Corp
1700 Main St
Santa Monica, CA 90406
(213) 393 0411 x427

RAND and Stanford have been running an ARPANET NCP frontend to their UNIX systems for over a year. Advantages are numerous: the frontend offloads this arduous task from the CPU. NCP hackers on the net know the kernel and ncpdaemon code of the original implementation are monstrous in size and complexity. After moving most of the NCP to the front end, your UNIX is left with a small, normal device driver that connects user-level processes with the network front end service via a simple, user transparent, front end protocol. The front end also supports multiple host connections and can be used to share one IMP port between several UNIX's.

The front end hardware is an 11/34 (or 11/23, 11/03) CPU connected to the host CPU via a high-speed point to point link. Currently tty lines are being used but shortly DMR-11 (at RAND) and DA-11 (at LBL, Mike Odell) links will be supported.

Porting UNIX to the IBM Series/1

T Heines, P Jalics
CS Dept Cleveland State University
Cleveland, OH 44115
(216) 687 4769

The authors have completed transporting Version 7 of the UNIX Operating System to an IBM Series/1 minicomputer. The design decisions in making use of the Series/1 hardware facilities will be discussed. The portability characteristics of the V7 system will be discussed as will the nature of the main difficulties encountered. The authors will propose a strategy for future transporting of UNIX.

This presentation was not only one of the most interesting but it was definitely one of the most enjoyable. The authors coined a new word to describe the PDP 11 byte swap problem: NUXI (pronounced like Nuke-See). The origin came from the first time they tried to boot the machine and looked at the first messages printed. Due to the NUXI problem, they discovered that file systems are not generally portable. As a result, special utilities had to be developed to generate Series/1 file systems on their PDP11. A suggestion made to future UNIX porters was for a compiler verification suite of test programs.

They also verified that the hardest part of the port was writing a new C compiler for the new machine. They were able to obtain a copy of a C compiler done at the University of Delaware, based on of the Ritchie C Compiler.

They have available a list of modules affected by the port and percentages of lines changed in each; also available to future porters are diff listings of their changes to point out the problem areas. The following percentages of lines were affected:

standalone	100%
commands	4%
kernel	25%
libc	37%
C compiler	61%
assembler	100%

They wrote 19000 total lines of new code.

A Device-Independent, Screen-Oriented Editor

Gary Aitken
OWL Associates
Box 2303
Grand Junction, CO 81501
(303) 245 2303

This editor is intended to be a word processing tool as well as a normal user's editor. While not a substitute for roff, nroff, and similar tools, it is viewed as a more manageable tool usable by secretaries, executives, documentation specialists, programmers, home computer users, and others who do not require all the complexities of nroff. The editor processes "clear text" files, preserving spatial relationships as they will appear in the final document; overstrike characters and backspace sequences are not distorted spatially when viewed on the display, but are none-the-less fully viewable and editable. Files may be sent directly to a hardcopy device, without the need of intervening filters. Multiple files of arbitrary size may be edited and merged. Special keystrokes have been kept to a minimum; the editor is designed to allow systems houses and end users to taylor these to their own needs. No software changes are required for different terminal types, or for applications using different keystroke maps. Written entirely in C, the editor is designed such that additional capabilities are easily added. It is also fully instrumented for rapid, accurate debugging.

A network independent message system

Karl Auerbach
INTERACTIVE Systems Corporation
1212 7th st
Santa Monica, Ca. 90401
(213) 450-8363

A new electronic mail system has been designed and implemented for UNIX and VMS. It is constructed upon a general mechanism for transporting files through a network. The transport system is capable of relaying files through intermediate hosts and over diverse types of telecommunications links. File delivery is made to an arbitrary set of programs selected by the addressee name. As such, the system is capable of operating simultaneously as more than simply an electronic mail system. The mail system permits transparent distribution of mail across host boundaries. In addition the mail system provides for delivery confirmations, hierarchical structuring of distribution lists, full name or title addressing, user profiles, and priority transmission.

The development of a new formatting package

Mark Kampe
INTERACTIVE Systems Corporation
1212 7th st
Santa Monica, Ca. 90401
(213) 450-8363

Nroff/troff provide a sufficiently skilled user with the ability to produce reasonably high quality output although at great cost in terms of CPU resources and often requiring non-negligible programming ability.

We have developed a formatting package called "ff" which is substantially smaller, faster, simpler to use and is capable of producing far more aesthetically pleasing output on a wide range of devices. The architecture which makes the performance, power and simplicity possible is radically different from that which underlies nroff and troff.

The presentation will give an overview of the architecture of the formating system (present and planned), a discussion of the motivations for that architecture and a comparison of this new package with other contemporary formatting systems (Tex & Scribe).

Terminal Independent CRT Software

Mark Horton
UC Berkeley CS Div
Berkeley, CA 94720
(415) 642 6258

The database used by the vi screen editor can be used by other programs as well. Since this database is widely available, there is no excuse for writing programs which only work on one terminal. A tour is conducted through the termcap database, the termlib routines for dealing with it, and the curses library for screen optimization. Several examples of programs using termlib and curses are presented.

Mark described the TERMCAP file of terminal characteristics and the terminal independent function library which uses that information. This software is available on both the previous 2BSD and the new 4BSD. Currently over 150 terminal types are supported and the number grows daily. Among the information that is parameterized in termcap is: size of screen, which capabilities exist and what strings are necessary to use them, how cursor addressing works, padding information, stty bits needed, initial string to be sent at login time, function keys, and provision for tender treatment of "brain-damaged" terminal designs.

A new higher-level function library called CURSES is now available, which is really an extraction of the IO routines in the vi editor. This library allows for optimal cursor movement and update of screen areas and handles multiple windows. It is almost as smart as vi although it lacks insert/delete line capabilities.

TROLL: a Compact Relational Data Base System

Anthony I. Wasserman
Medical Information Science
University of California, San Francisco
San Francisco, CA 94143
(415) 666-2951

TROLL is a relational database management system that was designed to provide the runtime database support needed by the programming language PLAIN, as well as to provide a mechanism for the use of alternate interfaces. The Data Base Handler was designed with the goals of modular architecture and compactness in mind. The highest level, the communication interface, accepts input in TROLL, an intermediate level language similar to the relational algebra. The next level is the relation operation level, then the storage operation level, and finally the pagehandler, which performs the Unix I/O operations. All relations are stored as simple prefix B-trees.

TROLL can be used as a standalone relational database system, or can be connected to other programs via the Unix pipe structure. TROLL runs as a SINGLE process on PDP-11 Unix Version 7 systems supporting separate i-and-d space.

A Database Application Design Environment

Marc Meyer
UC Berkeley
1617 Addison St
Berkeley CA 94709
(415) 843 1818

This is a Database Applications Design environment designed for Plantronics/Zehntel. It includes: (a) a screen formatting package. (b) a menu oriented interface to any and all utilities. (c) a relational database editor. (d) a simple report generation scheme for database systems.

INGRES: status and directions

Eric Allman
Project Ingres, Electronics Research Lab, Cory Hall
University of California
Berkeley, California 94720
(415) 642-2344

This talk/QA session will discuss the current status of the INGRES database management system and give approximations of future directions. The prepared material will be minimal, consisting of a quick overview of the versions currently or soon-to-be available and the research directions that we are embarking upon. Implementation would not be a subject of the prepared material. The majority of the session would be given over to answering questions.

Status of Ingres/6.3: This is the PDP11 version. It requires I/D space and FP and now runs on V7. It is functionally identical to 6.2. It is in final stages of release and will either go out directly (for a $150 tape copy fee) or through the next 2.xBSD tape. It is now in the public domain.

Status of Ingres/7: This is the VAX version. It runs as two processes (for protection reasons) and is about 2.5 times faster than the /70 6.3 version. It will be released in March on the 4BSD tape. It is also public domain.

Documents are available from Tandy Warnow at the above address, cash in advance. Introductory package $5 contains: Ref manual, tutorial, and design document (from TODS). The complete package of manuals (3 lbs.) is $30 and contains everything.

In the future the INGRES project will be moving "back to research" and will be less concerned with support of external users. The PDP11 version will be frozen and the VAX versions will not emphasize upward compatibility. Research topics to be explored include distributed databases, "hypothetical" databases, expert systems, and AI enhancements.

A Series of RDBMS Applications Using MRS

Christine M. Robertson
CSRG, University of Toronto
121 St Joseph St
Toronto, Ontario M5S 1A1 Canada
(416) 978 6060

MRS is a relational DBMS with a query language based on a subset of SEQUEL II, and runs under V6 or V7 UNIX. It was developed at the

CSRG, University of Toronto, in 1979, and has been distributed to over 50 sites worldwide. A series of MRS applications at the University are described in detail: PAY, a payroll authorization form storage and report generator system; STOCK, a system for keeping track of laboratory equipment; MES, a mark entry system for undergraduate courses; and CRABS II, a reference bibliography system with a user-friendly interface. The ease and speed of prototype applications using MRS is described, as well as mechanisms for improving speed and "custom" tailoring the front-end to user requirements. The control over accessibility of the database functions via custom-written C programs and shell routines, and the suitability of the Interactive Subsystem as a naive-user interface, are discussed.

Tools for Building an RDBMS

John Z. Kornatowski
Rhodnius Inc.,
P.O. Box 1, Station D,
Scarborough, Ontario Canada M1R 4Y7
(416) 922-1743

A set of data management tools using layered primitives has been created as a means of developing and enhancing database management systems. The higher level primitives enable features such as generalized index mechanisms, link and selector capabilities, and tuple-at-a-time operations to be implemented, resulting in efficient links, repeating groups, variable-length fields, and fully interactive database operations. With these tools, any of the three database models: Hierarchical, Network, or Relational, can be readily implemented. MISTRESS, a supported commercial relational database management system, has been developed using these primitives. MISTRESS implements a superset of the MRS query language (SEQUEL II-like) and is thus a user-oriented higher-level tool suited for application development. In addition, this layering makes MISTRESS potentially portable over a wide range of machines and systems. These implementation techniques have resulted in a state-of-the-art product that is highly flexible and easily enhanced.

Environmental Technical Information System

C. Corbin
USACERL-EN
PO Box 4005
Champaign, IL 61820
(217) 352 6511 x463

ETIS is a centralized system composed of several systems oriented toward the management of environmental information. It is used extensively by both the Department of the Army and the US Air Force, with

additional usage being made by a number of other federal agencies. The system consists of models, data bases, and management information programs tailored toward a pleasing and acceptable interface. The system is currently receiving in excess of 4000 retrievals from users this year and is being used by a number of different users. The system user is commonly a novice computer user, uninterested in the aspects of computer science, but more interested in the access and interpretation of data. The system consists primarily of the Environmental Impact Computer System (EICS), the Economic Impact Forecast System (EIFS), and the computer aided Environmental Legislative Data System (CELDS).

The system is maintained by the US Army Construction Engineering Laboratory. This research development agency is charged with research for the Chief of Engineers. As a federal R&D laboratory, other agencies have participated in the development of ETIS.

The DRAW Circuit Design System

Steve Bourne
Bell Telephone Labs
Murray Hill, NJ

The circuit design engineer who wants to obtain rapid prototype construction of one-of-a-kind designs can get help by using programs that run under the UNIX operating system. The programs include means for drawing circuit schematics on a graphics terminal and semi-automatic production of wire lists from those drawings. Included in the package are programs that check for simple errors and an interactive graphics program for performing the physical design of a circuit board.

The UCDS, Unix circuit design system, is an outgrowth of DRAW and other tools built by Fraser, Condon, Chesson, and Ditzel <see the article in the Aug 78 BSTJ>. A new capability recently added and used in the design of the "Belle" chess machine, is a "macro" facility. This allows replication of bussed signals to occur automatically and leaves circuit drawings much less cluttered-looking with only one line shown per group of bussed signals. Macros also allow the design to be organized hierarchically and eliminate the need to carry around extra hard-copy-only information such as block diagrams --the block diagram is now part of the machine readable description and boxes of it can expanded by pointing at them. The Condon and Thompson chess machine design takes only 25 drawings and makes much use of 64 bit busses. Condon has said that the design would have been impossible without this tool.

A new tool, PLH, does static design checking of parameters such power, hi/low current per pin, signal delays, dependencies, and setup times. The entire UCDS package has now been released for experimental use to MIT and may be released to other universities shortly.

UNIX as a Base for Large-Scale Applications

Michael D. Tilson
Human Computing Resources Corporation
10 St. Mary Street
Toronto, Ontario, Canada M4Y 1P9
(416) 922 1937

This talk is a case study of a large scale application based upon UNIX. In this context "large-scale" means a system connected to a network, with 60 simultaneous users accessing a several hundred megabyte database. The system is used in a telephone repair application, and requires both high performance and high reliability.

The methods used to meet these goals are described. This case study is of particular interest, because the application was already running under the manufacturer-supplied operating system. It was re-implemented under UNIX. The new implementation is more reliable, has higher performance, and is MUCH less expensive to maintain. Also, it is now possible to port the system to another machine.

This benchmark comparison will be helpful to those involved in the selection of operating systems to support large applications.

As a rough gauge of system complexity, the stack of line printer listings of the RSX based system was over 2 meters high, versus 3 inches for UNIX. Both systems were fully functionally equivalent. The previous system had an estimated maintenance cost of over $1 million per year and had taken 6 years to develop. The UNIX based system only requires a half-person per year for maintenance and was written in one year.

Compiler Error Analysis to Speed Program Debugging

Robert R. Henry
Computer Science Division
UC Berkeley
Berkeley, CA 94720
(415) 642-6258

The UNIX programming environment encourages users to quickly recompile and edit source programs to find and remove trivial syntactic and semantic errors found by the compilers. Often, there is no current listing for reference. During the early stages of debugging, the programmer is swamped with error messages, that are hastily jotted down in illegible handwriting in a secret, personal abbreviation system. The compilers produce error messages referenced to line numbers valid when the program was compiled, but invalid after lines are inserted into the source during an edit. Since most compilers on UNIX do not make a listing juxtaposing the program source and the errors it encountered, manipulating and remembering error messages can be frustrating.

Error is a program analyzing the error messages produced by a number of common UNIX compilers, including cc, lint, f77 and Berkeley Pascal. Error runs as an error message filter to the back end of a compilation run. Under user control, error inserts most of the error messages into the offending source programs after the line(s) the compiler complained about. Error messages are inserted as flagged comments, appropriate for the particular language. Some error messages can be discarded if the user has previously categorized them as noise. When error is finished, it runs an editor on the offending source files.

The author describes his system as a hacker's solution to one of the problems with hacking: compilation errors. This simple filter virtually removes the need to make a hardcopy listing of a program while you are trying to get the compilation time errors out. As the author point's out, if a compiler writer changes an error diagnostic, it breaks this program. But all things considered this is a very useful tool for adding a programmer.

A Pascal Compiler for the VAX

Peter B. Kessler
UC Berkeley Computer Science Division
Berkeley, California 94720
(415) 642 6258

A Pascal compiler has been constructed for the VAX by using the Berkeley Pascal interpreter and the second pass of the portable C compiler. The compiler retains the excellent error diagnostics and recovery of the Berkeley Pascal interpreter. The compiler supports the full Pascal language, including function and procedure parameters, and has been extended to provide separate compilation. Separate compilation does not sacrifice type safety between separately compiled portions of a program and allows Pascal routines to call and be called by routines written in other languages (C, F77, and Lisp, to date). The compiler allows debugging either with adb or sdb, the latter providing breakpoints, etc. with respect to the Pascal source statements.

This system is being distributed with 4BSD. It currently does not work on the PDP 11 because it actually uses the F77 back end. He also had to add a new pass to preform some in-line code expansion after the code generator. UCB may produce a version for other machines at a later date.

The author made an interesting note. He had intended to not have to learn VAX assembly language when he did this project. His hope for using another compiler's backend should have brought this goal to fruition. Unfortunately, he had to break down and learn the peculiarities of the VAX in order to finish this implementation.

Tcheck: a file system tree checker

John Thompson
University of Oklahoma
Norman, Oklahoma
(405) 325 4721

The topological properties of data structures may be represented by graphs or equivalently by K-formulas. Generators for these respective representations are graph grammars and K-formula schemata. It is known that the topology of the UNIX file system can be represented perspicuously using a K-grammar.

A recognizer, i.e., parser, based on a push down automaton (pda) is presented which "accepts" a UNIX file system iff it is structurally correct. The interesting parts of the C program "tcheck" (which is fundamentally a pda recognizer) are exhibited and the program's performance analyzed.

This method of file system checking is very mathematically based and has an interesting side effect. If the recognizer can be made to detect an error, it may be able to correct the error. At this time they do not have any bench marks to compare with Ted Kowalski's fsck program, but they believe it runs as fast.

Putting Writing Courses Online With UNIX

James Joyce
International Technical Seminars
47 Potomac Street
San Francisco, CA 94117
(415) 621 6415

Teaching writing, whether to freshman students, juniors who need to pass a writing proficiency exam, or adults in courses on effective writing, is a difficult problem that seems to resist any solution. The magnitude of the problem is measured by the sheer numbers of people involved. Nearly everyone who spends any time attending an institution of higher education takes a writing course, whether or not that person graduates. This number is clearly a least upper bound, as it does not count the number of people who take courses on effective writing in university extension programs or from private, seminar-giving concerns.

With some notable exceptions, attempts to bring computer technology to bear on the teaching of writing has been mired in CAI drills designed to test spelling, usage, or logic --which are parts of, but not the totality of writing. Writing is made up of several related parts, and this presentation will show how UNIX can help writers with: (1) writing something that can be considered a first draft; (2) analyzing what is there for correctness and appropriateness; (3) refining the draft into final form.

Further, teaching writing imposes other requirements UNIX can help with: (4) students need to be able to communicate problems they are having with each other or the teacher; (5) the teacher needs to be able to evaluate the written work.

Where many text editors and formatters are either underpowered (such as that on the Apple) or expensive (such as WYLBUR on IBM mainframes), UNIX is both powerful and inexpensive, particularly in its recent microcomputer implementations.

UNIX/TOOLS macro package

Glenn D. Watt
Air Force Data Service Center
Pentagon Bldg
Washington, DC 20330
(202) 695 7904

This paper covers a description of a new general purpose formatting language called UNIX/TOOLS. The language was developed from a sophisticated set of nroff macros <PWB mm macros> along with numerous enhancements to the nroff/troff formatters.

Glenn has rewritten the PWB MM macros package to run much faster by only loading those macros that are need at any given time. He has modified nroff/troff for their needs. He combined many earlier macro packages into one all encompassing version that could tailor itself to the vast number of a different documents used in the Pentagon. One of the major additions that he has added, is a scheme for handling two column output correctly at end of page. The current nroff/troff scheme will create one long column followed by one short column. Glenn's version will even these two columns up to make both columns approximately the same size.

Description of a file locking scheme implemented for UNIX V7; and Hardware/Software design issues for UNIX on small systems.

John L. Bass
ONYX Systems Inc.
73 East Trimble Road
San Jose, CA. 95131
(408) 946 6330

UNIX has a design feature which allows multiple processes to have read/write access to the same file. This locking scheme allows full use of that feature by providing controlled access to file segments which are to be updated. The locking scheme allows multiple variable length segments in open files to be access locked to other processes. The implementation provides for automatic lock removal on close or process termination. A key feature of this locking scheme is the detection of deadlock conditions between processes controlling locked file segments. The locking scheme requires several minor changes to inode.h, fio.c, sys2.c and sysent.c to install. The sources and implementation description are in the public domain, and are available from the author.

UNIX on smaller low cost machines is subjected to limitations not normally found on larger machines. Further more the cost performance tradeoffs are sufficient to make normal selection decisions difficult. The presentation will include discussion of performance issues relating to serial IO, disk I/O, and memory utilization.

John gave a interesting talk pointing out a major flaw in the UNIX I/O system. Some languages, namely COBOL, which uses ISAM files need some sort of file locking in order to keep two processes from wiping each other's data out of a file. The ONYX machine has implemented a new system call to allow two or more processes to update a file with full read/write access without trashing the file or deadlocking any of the processes. In the name of portability, ONYX has generously offered their solution to everyone in the UNIX community. Write to ONYX about the availability of the code and the documentation for it.

Next John spoke about some problems that ONYX ran into in regards to TTY I/O when they brought up the ONYX machine. The ONYX machine uses Zilog SIO chips to provide the serial RS-232C interface. Like many simplistic RS-232C interfaces this is not a DMA interface and ONYX quickly discovered that the CU (call Unix) program was not reliable at baud rates above 300 baud. After many hours of debugging they realized the interrupt overhead for character by character I/O was too great. DEC interfaces like the DH-11 and the DZ-11 have a silo, so that if you do not respond to the interface in one character time, the characters are buffered in a hardware silo. When UNIX actually reads the DH, it gets multiple characters instead of just one. This lessens the per character response time for any given character.

Also, ONYX discovered that although "Winchester" disks have fast transfer times, they seek very slowly. ONYX has done some work in their version of UNIX to lessen the load on the disk at any given time to get their average disk latency's back up to the level that the RM and RP have on the 11/70's.

UNIX software and hardware from Bedford Computer

Eric Woudenberg
Bedford Computer
Tirrell Hill Road
Bedford NH 03102
(603) 668 3400

"PDP-11 UNIX is out of gas," to quote a Bell Labs lecturer. Much of V7 UNIX's speed problems on the 11/34 and 11/45 arise from insufficient physical memory address space. A hardware device built by Bedford to alleviate this will be explained as well as the modifications we made to UNIX to use it.

Debugging the UNIX kernel has (as far as I know) always been a tiresome process, involving dumps or extensive switch register sessions. Bedford has developed (from an earlier pdp-11 ddt) a UNIX kernel ddt. This allows symbolic instruction typein/typeout as well as breakpoints and single stepping of the kernel. There are versions of this available for use with and without our special hardware.

Bedford has produced a PC Board the plugs into a PDP-11 that will give you the 22 bit mode and UNIBUS Map from SSR3, one of the two missing kernel registers from the PDP 11/40 class processors (11/40, 34, 35, 23). This board will allow a PDP 11 to address 4 MByte like a PDP 11/70 and a PDP 11/44. It will not have ever give you the separate I/D space bit in SSR0 or the fault register (SSR1) unless you already have it (PDP 11/45).

Vendors Announcing UNIX related products and services

Typeset direct from your UNIX system

George Lithograph
650 Second Street
San Francisco, CA 94107
ATTN: Len Schafer
(415) 397-2400

Local Network for Multivendor Distributed Processing

Ungermann-Bass, Inc.
2560 Mission College Boulevard
Santa Clara, CA 95050
(408) 496-0111

PDP 11/23 based DYNASTY System 20 running DYNIX.

The Santa Cruz Operation Inc.
500 Chestnut Street
Santa Cruz, CA 95060
(408) 425-7222

8,000 Character Display Terminal

Sierra Information Machines
4676 Admiralty Way, Suite 226
Marina del Rey, CA 90291
(213) 823-8203

The C Machine - Microcoded computer system designed for C.

BBN Computer Corporation
33 Moulton Street
Cambridge, MA 02238
(617) 497-2000

Interactive ONYX

ONYX systems running IS/1 connected via an Ungermann-Bass network.

Interactive Systems Corporation
1212 Seventh Street
Santa Monica, CA 90401
(213) 450-8363

Concept Series Display Terminals

Human Designed Systems, Inc.
3700 Market Street
Philadelphia, PA 19104
(215) 382-5000

ISO Pascal to C translator for PDP-11

Whitesmiths, Ltd.
POB 1132 Ansonia Station
New York, NY 10023
(212) 799-1200

Coherent Operating System

UNIX-compatible operating system for PDP-11/34

Mark Williams Company
1430 W. Wrightwood Avenue
Chicago, IL 60614
(312) 472-6659

Mistress-Relational Database Management Systems

Rhodnius, Inc.
P.O. Box 1, Station D
Scarborough, Ontario
Canada M1R 4Y7
(416) 922-1743

Bulletin Board

The following items were derived from notices posted on the bulletin board at the San Francisco Usenix Conference.

* Anyone using a Hewlett Packard 4-color Plotter?

Kate Rosenbloom
MS 239-19
Ames Research Center
Moffett Field, CA 94039
(415) 965-6436

Yes,M. Gates, Johns Hopkins/APL, Laurel, Maryland

Yes, Mike Muuss, BRL-US Army, MMUUSS@mit-mc

* I am interested in Circuit Design (Analog & Digital) Software that runs on UNIX.

Ron Birchall
Tellabs
(312) 979-8800

I am too:

Scott Bertilson
Rosemount Inc.
(601) 456-1121
have el-cheapo ECAP running.

* HELP! C compiler for a SEL or (better yet) a UNIX that runs on a SEL32? Who has one? Contact:

Walt Donovan
Ames Research Center
Mail Stop 240-8
Moffet Field, CA 94035

* MINC-11 or LSI with IEEE-488. Anyone who has UNIX software for lab data acquisition and device control, please Contact:

Gary Newman
Ampex Corporation
(415) 367-3911

- In place File System Conversion V6 to V7. Contact:

  Dean Johnson
  Univ of Minnesota
  Dept. of Mech. Eng., Heat Transfer Division
  238 M.E.
  Minneapolis, MN 55455
  (612) 376-2629

- I would like to communicate with anyone with experience with handlers for:

  DeAnza Display System
  RL01 Disc Drives
  DR11 and DRV11 (B) interfaces.

  We currently run V7 UNIX on a PDP 11/45 and V6 Unix on an LSI 11/23 (RK05 discs):

  Dennis Parker U.C.S.F
  M330
  U.C.S.F
  San Franicisco, CA 94143
  (415) 666-1208

- Who has information about an RJE link to

  - Siemens 7760 running under BS2000

  - ICL 2980/82 under VME/B

  - ICL 2976 under VME/B

  Please contact:
  Wilfried De Meyere
  Bell Telephone Mfg. Co./ITT
  F. Wellerplain 1
  2000 Antwerp
  Belgium

* Interested in talking with anyone using UNIX and "C" for management information type applications development. Also information on products such as project management (CPM), statistics, etc., available for PDP 11/70 - UNIX Version 7.

Linda Wilson
CYLIX Communications Network
Memphis, TENN
(901) 761-1177

Yes: Rudy Ramsey
(ITT PTC)

Scott Bertilson
Rosemount, Inc
(612) 937-3602, 456-1121

John Copeland
Star Rt. Box 262
Muir Beach, CA 94965
(415) 383-1281

Have any? Contact:
D. Fiedler
(201) 533-0405

* Need info on C for HP-3000 (and Burroughs B1800) Contact:

Jay Lepreau
lepreau@utah-20
CS Dept, MEB.
Univ. of Utah
Salt Lake City, UT 84112
(801) 581-8322

* I have HP3000 "C" compiler (not quite finished, need help):

Larry Bakst
g.leb@mit-ee
or
16 Country Club Dr., Apt 37
Manchester, NH 03102
(603) 624-4242
(603) 668-3400 (work)

* Need info on UUCP:

1. Bringing it up on V6 UNIX.

2. Using Vadic Autodialers

3. Info on the hardware connections.

* Have UUCP on V6.

  Mike Muuss
  BRL
  (301) 28-5691
  muuss@darcom-ka
  or
  talk to
  Peter Gross @HAO

* Like to bring up UUCP on 11/70 machines.

  Fran Chan
  Logicon-Intercomp
  (213) 325-6060

* Does anyone have reliable diagnostics for the Systems Industries 9500 Controller on the VAX? (617) 944-6850, Carol Gross.

* HP1000

  Want C Compiler and UNIX. Anyone with info or interested, please leave name and number (phone) below.

  Scott Bertilson,
  Rosemount, Inc.
  (612) 456-1121

* Would like to contact anyone using

  * Bunker Ramo disks (BR 1538C)

  * UNIX Cobol compiler

  Bob Domey
  Mitre Corp.

  For the former, pester Bunker Ramo ESD in Westlake, CA

* We are interested in contact with other UNIX people concerning

  * Using CC 9766 Disk (Emulex Controller)

  * PDP 11 interface with IBM/34 (RJE)

  * CAD - Automated Printed Circuit Design on UNIX

  * Circuit Analysis Program (we have a simple one in Fortran)

  * Micro-Processor Tools both 16 bit and 8 bit Micros specially C, M6800, Dynamic Debugging Tools, and Portable Real Time OS on 16 bit micros.

Please contact:
Chia-My Chang and Mike Green
Emerson Electric Co.
Doric Scientific Div.
3883 Ruffin Road
San Diego, CA 92123
(714) 565-4415

* SATELLITE PROCESSOR SYSTEM For information:

Charlie Price
Storage Technology Corp.
2270 S. 88th Street
Louisville, CO 80027

It should be available NEXT meeting. BUT. If someone has a use for such a thing immediately and will bring it up in the next six months as a test site I'd be willing to make it available for shaking bugs out.

This runs on a PDP 11 under version 7 and support LSI-11 satellites. We use a serial link (DLV11 type).

* Looking for People Interested in "C" Compiler and Portable Assembler & Tools for M68000 (MIT and others).

Pete Delaney
(303) 773-4700
Carlos Clarke
(401) 847-8000, ext. 3105

PS. Any Interest in UNIX on Honeywell Level-6 will also be entertained.

* I am interested in the following:

1. Exchange information on UNIX V7 (maintenance, problems, ideas, etc.)

2. Information on C compilers for MC68000.

3. Information on PERT programs on UNIX

Please contact:

Ping K. Liao
Advanced Micro Computer
(408) 988-7777, ext 297

Response to #3: M. Moussavi, (202) 676-9087

* PL/I on VAX/UNIX

I would appreciate knowing of any available or planned PL/I compilers for VAX/UNIX (4 BSD).

Paul Messina
Argonne National lab
Argonne, IL 60439
(312) 972-7162
messina@lbl-unix
messina@BBNC

* KMC-11 support needed:

KMC-DUP11 controller
KMC cross assembler
HASP protocol

Al Segal
Cray Laboratories
5311 Western Avenue
Boulder, CO 80301
(303) 449-8470

* 8086 C

If you have any information on a cross compiler running under Version 7 UNIX, please give me a call.

Steve Beisner
Com Design
Goleta, CA 93017
(805) 964-9852

Yes, Ken Brown, (604) 294-0414
<See also p. 7 -- WW>

* Harris Phototypesetter

1. We are going to feed a HARRIS phototypesetter with output from nroff and special developed formatting programs. Has anybody experience with feeding a Harris machine?

2. EUUG meeting (European UNIX User Group) Monday, April 13, 1981, Math Center, Amsterdam

Leus Hagen
Math Centre
Kruislaan 413
1098 SY Amsterdam
Netherlands
(20) 592 9333

* Graphics On UNIX

We are undertaking both high-performance and batch graphics on a UNIX system. We are interested in exchanging information with others who know about:

- * Megatek graphics systems on UNIX.
- * SIGGRAPH Core packages operating under UNIX.
- * Serious UNIX-based graphics efforts in general.

Please contact:
Rudy Ramsey
ITT Programming Technology Center
1000 Oronoque Lane
Stratford, CT 06497
(203) 375-0200

* I am interested in STATISTICAL PACKAGES running under UNIX, especially ones designed for ECONOMICS applications. Any information would be appreciated.

Chris Robertson
CSRG
University of Toronto

* Disc Driver

We have a driver for the MCT SMC-11/EDC-21 disc controller which handles:

1. overlapped seeks
2. multiple controllers
3. ECC/offset recovery
4. BAD BLOCK REPLACEMENT

Bad block replacement requires a modified formatter (supplied), which writes a DEC-style bad block track on the last track of the disk.

Currently for Version 6 only (interested in V7 test systems).

Harold Selbrig
MCT
2470 Embarcadero Way
Palo Alto, CA 94303
(415) 856-7400

* The Rand Corporation

If you've had trouble receiving software from Rand or, if you would just now like to receive software from Rand: Take heart! All procedural problems have been resolved. Contact Michael Wahrman, (213) 393-0411 or Mike@rand-unix. I'd particularly like to talk to anyone who thinks they should have received a tape, and hasn't.

* Tools on RT-11? If you have info please call:

Garret Tollkuhn
Clark Kerr Hall
University of California
Santa Cruz, CA 95064
(408) 429-4457
(408) 429-2900 and leave message.

* Michael Bloom of California State University, Northridge is looking for advice or help in getting a Version 7 UNIX system running on his system. The machine is a PDP-11 of some appropriate form with a rather unusual complement of disk and tape devices. His disk is a System Industries, sort-of-RM03 look-alike, but with 33 sectors/track rather than 32. His tape is a 7-track Cypher 900X (7-track TU-10 look-alike). CSUN has facilities to translate the 9-track Bell distribution tape into various forms of 7-track format. If you think you may be able to help, call:

(213) 354-5543 (office)
(213) 349-8136 (home)

The Usenix Association

PURPOSE: The Usenix Association is an organization of Western Electric licensees and sub-licensees formed for the purpose of exchanging information and ideas about the UNIX operating system and the C Progamming Language.

MEMBERSHIP: Four classes of membership in Usenix are offered:

1. Institutional Membership. Institutional Members are the voting members of the Usenix Association. This class of membership is open only to licensees or sub-licensees of Western Electric Co.

2. Non-voting Institutional Membership. This class of membership is open to corporate affiliates of AT&T.

3. Individual Membership. Open to employees of class 1 and 2 members and others who are bound by the software agreements with Western Electric and its licensees.

4. Public Membership. Open to anyone with a bona fide interest in the purpose of the Usenix Association.

For further information write:

Usenix Association
Rockefeller University
Box 8
1230 York Avenue
New York, New York 10021
(212) 360-1182

Facts about UNIX and the Programming Language C

The UNIX operating system was developed by Ken Thompson and Dennis Ritchie of Bell Laboratories in Murray Hill, N.H., during the early 1970's. The C Programming Language was developed originally by Thompson and Ritchie as the implementation language for UNIX. UNIX is made available to the public by Western Electric Co. through its patent licensing office in Greensboro, North Carolina.

A fine overview of UNIX and C was published in the Bell System Technical Journal, Vol. 57, No.6; Part 2, in August 1978. The C Programming Language is described in the book _The C Programming Language_ by Brian Kernighan and Dennis Ritchie published in 1978 by Prentice Hall.

Login Newsletter V6N1
Computation Center
University of Texas at Austin
Austin, Texas 78712